(19)日本国特許庁（ＪＰ）　(12) **公開特許公報**（Ａ）　(11)特許出願公開番号
特開2000－67033
（P2000－67033A）
(43)公開日　平成12年３月３日(2000.3.3)

(51)Int.Cl.[7]	識別記号	Ｆ Ｉ	テーマコード（参考）
Ｇ０６Ｆ　17/18		Ｇ０６Ｆ　15/36	Ｚ

審査請求　未請求　請求項の数10　ＯＬ　（全 11 頁）

(21)出願番号　特願平11－148862

(22)出願日　平成11年５月27日(1999.5.27)

(31)優先権主張番号　９８４０１２６７．４

(32)優先日　平成10年５月27日(1998.5.27)

(33)優先権主張国　ヨーロッパ特許庁（ＥＰ）

(71)出願人　599072932
ソニー　フランス　エスアー
フランス国　75831　パリ市　セデックス　17番フロレアル通り　15番地

(72)発明者　パシェ、フランソワ
フランス国　75005　パリ市　アミヨ通り　６番地ソニー　コンピュータ　サイエンス研究所　パリオフィス内

(72)発明者　ロイ、ピエール
フランス国　75005　パリ市　カルディナル　ルモワンス通り　83番地

(74)代理人　100067736
弁理士　小池　晃　（外２名）

(54)【発明の名称】　シーケンス情報生成方法及びシーケンス情報生成装置

(57)【要約】
【課題】　各アイテムの類似性や相違性を考慮し、所望の基準に適合するアイテムのシーケンスを効率的に生成する。
【解決手段】　データベースに登録された各アイテムを変数とし、所望のシーケンスの特性を制約として、シーケンス生成問題を制約充足問題として定式化し、これを解決する。

【特許請求の範囲】

【請求項１】　データベースから複数のアイテムを選出し、これら複数のアイテムから構成されるシーケンスを示すシーケンス情報を生成するシーケンス情報生成方法であって、
複数のアイテムの属性を示すデータが登録されたデータベースを準備するステップと、
所望のシーケンスにおけるアイテムの属性の値又は値の変域に対する要求により所望のシーケンスの特性を特定するステップと、
上記所望のシーケンスにおける各アイテムを変数とし、上記所望のシーケンスの特性を特定するステップにおいて特定された特性を制約とすることにより、所望のシーケンスの生成を制約充足問題として定式化するステップと、
制約充足問題プログラミング技術を用いて、上記定式化された制約充足問題を解決するステップとを有するシーケンス情報生成方法。

【請求項２】　上記データベースには、異なる値間の類似及び相違を定義する所定の分類法により分類された属性を少なくとも１つ有するアイテムが格納されており、上記定式化を行うステップにおいて、上記所望のシーケンスにおける異なるアイテム間の類似又は相違を要求する少なくとも１つの制約を生成するステップを有することを特徴とする請求項１記載のシーケンス情報生成方法。

【請求項３】　上記定式化を行うステップにおいて、上記所望のシーケンス内で連続する複数のアイテムの一部の隣接するアイテムに関する類似又は相違を要求する少なくとも１つの制約を生成することを特徴とする請求項２記載のシーケンス情報生成方法。

【請求項４】　上記定式化を行うステップにおいて、上記所望のシーケンス内で連続する全てのアイテムに関する類似又は相違を要求する少なくとも１つの制約を生成することを特徴とする請求項２記載のシーケンス情報生成方法。

【請求項５】　上記定式化を行うステップにおいて、上記所望のシーケンス内で属性が所定の集合に属する値を有するアイテムの数が特定の範囲内でなくてはならないことを要求する少なくとも１つの制約を生成することを特徴とする請求項１乃至４いずれか１項に記載のシーケンス情報生成方法。

【請求項６】　上記定式化を行うステップにおいて、上記所望のシーケンス内で連続するアイテムの一部において、属性が所定の集合に属する値を有するアイテムの数が特定の範囲内ではなくてはならないことを要求する少なくとも１つの制約を生成することを特徴とする請求項５記載のシーケンス情報生成方法。

【請求項７】　上記定式化を行うステップにおいて、複数のアイテムの属性に関する相違する値の数が特定の範囲内でなくてはならないことを要求する少なくとも１つの制約を生成することを特徴とする請求項１乃至６いずれか１項に記載のシーケンス情報生成方法。

【請求項８】　上記定式化を行うステップにおいて、上記所望のシーケンスにおける連続する一部のアイテムにおいて、これらアイテムの属性に関する相違する値の数が特定の範囲内でなくてはならないことを要求する少なくとも１つの制約を生成することを特徴とする請求項７記載のシーケンス情報生成方法。

【請求項９】　上記データベースは、楽曲データを格納するデータベースであり、上記シーケンス情報は、上記データベースから選択された楽曲の再生シーケンスを定めるリサイタル情報であることを特徴とする請求項１乃至８いずれか１項に記載のシーケンス情報生成方法。

【請求項１０】　上記請求項１乃至９のいずれか１項に記載のシーケンス情報生成方法を実現するシーケンス特性特定手段と定式化手段と解決手段とを備える汎用コンピュータと、
上記シーケンス情報生成方法に基づいて生成されたシーケンス情報を表示する表示手段とを備えるシーケンス情報生成装置。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】本発明は、シーケンス情報生成方法及びシーケンス情報生成装置に関し、特に各構成アイテムの属性間の関係を管理しながらシーケンスを生成するシーケンス情報生成方法及びシーケンス情報生成装置に関する。

【０００２】

【従来の技術】複数のデータからなるコレクションの内から、特定のデータをいくつか抽出し、抽出したデータを一定の順序で配列して、所望の条件を満たすシーケンスを生成する必要があることがある。この場合、シーケンスが「コヒーレント」である、すなわち、各構成アイテムの属性の間に特定の関係が存在することが求められることがある。例えば、シーケンス内において連続するアイテムは、互いに類似していなくてはならない、あるいは、同一であってはならない、等の関係が求められることがある。なお、ここでいうシーケンスという用語には、時間的シーケンスと空間的シーケンスとが含まれる。このような関係が求められる場合としては、例えば、マルチメディアの分野において、特に時間的シーケンスとしてリサイタルを自動的に生成する場合が挙げられる。ここで「リサイタル」という語は、単に楽曲のシーケンスを指すのみでなく、より包括的な意味として映像クリップや記録データ、テキストアイテム等のマルチメディアのアイテムのシーケンスを指す。なお、空間的シーケンスを生成する場合とは、例えば画廊の空間内で各作品を主題別に関連付けてレイアウトを決定する場合などがある。

【０００３】
【発明が解決しようとする課題】１９９８年に、ベルカウイ（Berkaoui）他共著、「制約による論理プログラムに関するジュルネ　フランコフォン（Journees Francophones de ProgrammationLogique par Contraintes）」中の文献「部分PQR階層構造によってインスタンス化されていない集合上に定義されたシーケンスの表現”Ｒｅｐｒｅｓｅｎｔａｔｉｏｎ　ｄｅ　ｓｅｑｕｅｎｃｅｓ　ｄｅｆｉｎｉｅｓ　ｓｕｒ　ｄｅｓ　ｅｎｓｅｍｂｌｅｓ　ｎｏｎ　ｉｎｓｔａｎｃｉｅｓ　ｐａｒ　ａｒｂｒｅ　ＰＯＲ　ｐａｒｔｉｅｌ”」において、スケジューリングの問題を解決するため制約充足問題（ＣｏｎｓｔｒａｉｎｔＳａｔｉｓｆａｃｔｉｏｎ　Ｐｒｏｂｌｅｍ、以下ＣＳＰという。）を用いることが提案されている。しかし、この文献において提唱されている手法では、生成されるアイテムのシーケンスの「コヒーレンス性」が考慮されておらず、すなわち、シーケンスを構成するアイテムの類似性或いは相違性といった関係を管理することができない。
【０００４】そこで、本発明は、上述の課題に鑑み、複数のアイテムから構成されるシーケンスの「コヒーレンス性」を考慮し、シーケンスを形成するアイテムの類似性又は相違性といった関係を管理できるシーケンス情報生成方法及びシーケンス情報生成装置を提供することを目的とする。
【０００５】
【課題を解決するための手段】上述の目的を達成するために、本発明に係るシーケンス情報生成方法は、データベースから複数のアイテムを選出して、これら複数のアイテムからなるシーケンスを示すシーケンス情報を生成するものであり、複数のアイテムの属性を示すデータが登録されたデータベースを準備するステップと、所望のシーケンスにおけるアイテムの属性の値又は値の変域に対する要求により所望のシーケンスの特性を特定するステップと、所望のシーケンスにおける各アイテムを変数とし、所望のシーケンスの特性を特定するステップにおいて特定された特性を制約とすることにより、所望のシーケンスの生成を制約充足問題として定式化するステップと、制約充足問題プログラミング技術を用いて、定式化された制約充足問題を解決するステップとを有する。
【０００６】また、本発明に係るシーケンス情報生成装置は、例えば汎用コンピュータにより実現され、所望のシーケンスにおけるアイテムの属性の値又は値の変域に対する要求により所望のシーケンスの特性を特定する特定手段と、所望のシーケンスにおける各アイテムを変数とし、上記所望のシーケンスの特性を特定するステップにおいて特定された特性を制約とすることにより、所望のシーケンスの生成を制約充足問題として定式化する定式化手段と、制約充足問題プログラミング技術を用いて、定式化された制約充足問題を解決する解決手段と、生成されたシーケンス情報を表示する表示手段とを備える。
【０００７】本発明に係るシーケンス情報生成方法及びシーケンス情報生成装置は、特定の順序に並べられたアイテムの「コヒーレント」なシーケンスを生成するシステム及び方法を提供する。この場合のアイテムは、通常、データベースに格納されており、属性を示すデータと属性の値を示すデータの組として記述される。
【０００８】なお、「データベース」という語は、すでに格納されたデータ及び順次格納されるデータの両方を含めたデータのコレクションを指すものとする。所望のシーケンスを生成する際の問題は、ＣＳＰとして処理される。生成されるシーケンスは、データベース内のアイテムに固有の制約の集合を定式化することによって特定され、それぞれの制約は、シーケンスの特定の特性を示す。
【０００９】本発明において、データベース内のアイテムは、少なくとも幾つかの属性値による分類を伴った特定の汎用フォーマットを有する。また、制約は、既存の、特別に定式化された制約クラスのライブラリのなかから特定される。本発明ではこの特別な制約によって目的のシーケンスの所望のプロパティ、特に一連のアイテムの類似性や相違性を指定することができる。これらの制限条件クラスにより、コヒーレントなシーケンスのプロパティを簡潔に表現することができる。
【００１０】本発明の実施の形態では、データベース内の複数のアイテムに対応する汎用フォーマットと特別な制約クラスを用いることにより、ＣＳＰの手法を用いて複数の制約を同時に満たす一定の順序に配列されたアイテム、すなわちシーケンス情報を生成することができる。
【００１１】
【発明の実施の形態】以下、本発明に係るシーケンス生成方法及びシーケンス生成装置について詳細に説明する。
【００１２】以下の説明では、ラジオや有線放送の音楽番組などの音楽リサイタルを自動的に構成する場合を例に説明する。しかし、本発明は、このような形態に限定されず、様々な分野に応用されるものである。
【００１３】以下の記述では最初に、好ましいデータベースのフォーマットと、問題をＣＳＰとして定式化する手法につて説明し、さらにリサイクル構成問題をＣＳＰに適用する場合に用いられる特別な制約クラスについて解説した後、どのようにしてこの問題が定式化され、特別な制約クラスに属する制約を用いて解決されるかについて説明する。
【００１４】データベース
各アイテムは、属性を示すデータと値を示すデータの組により記述される。この実施例において音楽リサイタルを構成するアイテムは、音楽作品であり、インデック

ス、すなわち楽曲の固有識別子、曲名、アーティスト、演奏時間、形式、主要楽器、オーケストレーションの型、その他の属性を有する。

【００１５】これらの属性の値は、離散的な値、例えばボイスタイプや音楽スタイルといった、所定の存在し得る値の集合に属するものであってもよく、あるいは、例えば演奏時間等の連続的な値であってもよい。なお、音楽スタイルの分類法等については、発明の趣旨に直接関わるものではないため、ここでは詳細には説明しない。

【００１６】また、集合的又は複合的なパラメータ、すなわち他の上位のパラメータの構成要素となっているパラメータについて考慮に入れることもできる。これは、主要原理を単に拡張したものであり、このことによりいかなるパラメータの組み合わせに対しても制約を設定できるようになる。例えば、本実施例においては、データベースには、「ファミリー」及び「サブファミリー」という２つのパラメータを設けており、それらを組み合わせることで、「ポップ」や「ロック」のような「スタイル」が表示できる。

【００１７】データベース内に格納されたアイテムは、データベース上の該当するアイテムのインデックスをｉとしてP_iと示され、また、属性は、属性のインデックスをｊとして、$P_i a_j$と示される。通常、第１の属性はアイテム自身のインデックスである。

【００１８】ＣＳＰの定式化

本発明では、アイテムの集合、特に時間的及び空間的シーケンスに関する問題を制約充足問題（Constraint Satisfaction Problem：以下ＣＳＰという。）として定式化し、解決する。この定式化は、集合を限定された変数として示すことによって行われる。各変数はデータベースのアイテムの集合全体からなるドメインを有する。例えば、楽曲１０曲分のシーケンスは１０の制限された変数v_1，v_2・・v_{10}として、すなわち、第１のアイテムはv_1、第２のアイテムv_2等と表される。

【００１９】生成するシーケンスのコヒーレンスは上述の変数に対応した制約によって表される。時間的集合の＊生成は、汎用ソルバーによって行うことが望ましく、このようなソルバーは、制約充足アルゴリズムを利用するものである。

【００２０】シーケンスのプロパティを記述する制約は、以下の通りである。

【００２１】１）ＣＳＰの手法による対応が可能な、基本的で、単純な制約。例えばシーケンスの総継続時間を１時間ちょうどに指定するような制約は、一次結合式（LinearCombination）によって、また１時間以上や１時間以下に指定するような制約は、一次不等式(Linear Inequality)等の線形演算式によってそれぞれ具現された制約である。

【００２２】２）以下に示す特定の制約クラスを用いた制約。

【００２３】特定の制約クラスを用いた制約

この実施の形態においては、それぞれ包括的な数式で表すことができる基礎制約（cardinality constraints）、類似制約（similarity constraints）、相違制約（dissimilarity constraints）といった３つの特別な制約を定義する。なお、後述するように、類似制約と相違制約とを一括する「ランニング制約」といった定義も用いる。

【００２４】まず、基礎制約について説明する。この制約クラスでは、アイテムの集合に対してそのプロパティを指定する。基礎制約は、アイテムに関する基礎制約と属性値に関する基礎制約とに分類される。

【００２５】アイテムに関する基礎制約

この制約クラスでは、属性ｊが所定の集合Ｅに属しているアイテムの数が［ａ，ｂ］の範囲内でなくてはならないことを宣言することができる。この制約は、アイテムのインデックスをＩとし、属性のインデックスをｊとし、ａとｂを整数とし、属性ｊがとりうる値の部分集合をＥとして、以下のように表現することができる。

【００２６】

【数１】

$$CI(I,j,a,b,E) \text{ すなわち } \mathrm{card}\{i\in I \mid p_i a_j \in E\} \in [a,b]$$

【００２７】リサイタルの構成を決定する場合、この制約は、例えば最初の１０曲まで等の所定の範囲内において、音楽スタイルが「ロック」である楽曲の数が４～６の範囲内でなくてはならない、といった条件の宣言に用いることができる。この制約は、データベース内の楽曲のスタイルに対応したパラメータのインデックスをｓとして、CI({1,2,...,10},s,4,6,{"Rock"})と表現することができる。

【００２８】属性値に関する基礎制約

※この制約クラスでは、属性ｊが所定の集合Ｅに属しているアイテムの数が［ａ，ｂ］の範囲内でなくてはならないことを宣言することができる。この制約は、アイテムのインデックスをＩとし、属性のインデックスをｊとし、ａとｂとを整数として、以下のような数式で表現することができる。

【００２９】

【数２】

$$CA(I,j,a,b) \text{ すなわち } \mathrm{card}\{p_i a_j \mid i\in I\} \in [a,b] \quad ※$$

【００３０】例えば、リサイタルの構成を決定する場合、この制約は、最初の３曲が少なくとも２種類の、テンポが異なる曲である、すなわち、同じテンポであってはならないということを宣言するために用いることがで

きる。この制約は、データベース内の該当するテンポに対応するパラメータのインデックスを表すｔを用いてCA({1,2,3},t,3,3)と表現することができる。

【００３１】この属性値の基礎制約を用いて、包括的な相違性を宣言することもできる。インデックスの集合をＩとし、パラメータのインデックスをｊとして、包括的な相違性に関する制約Ｄ（Ｉ、ｊ）は、集合Ｉ内の各変数X_iの値をp_iとして、以下の関係が成立することを示す。

【００３２】

【数３】

$$\forall k \in I, \forall l \in I, p_k.a_j \neq p_l.a_j$$

【００３３】すなわち、この式に含まれる全ての変数は、属性ｊに関して、それぞれ一対の異なる値を有するということである。この制約は、以下のような特定の基礎制約として宣言することもできる。

【００３４】

＊S(a,b,j similar(.,.,.))means that:

for each item p_i,i∈[a,b-1]I,the predicate similar(p_i,p_{i+1},j)is true,

（各アイテムp_iについてi∈[a,b-1]Iが成り立つとき、述部similar(p_i,p_{i+1},j)は真である）

【００３８】なお、ここで、similar(.,.,.)は述語であり、また、述語の最初の２つのパラメータは楽曲であり、３番目のパラメータは、パラメータのインデックスである。

【００３９】リサイタルの適用例においては、この制約は、例えば１番目から１０番目までといった所定の連続した範囲内にある全ての楽曲が類似したスタイルを有していなくてはならない、といった条件の宣言に用いられ※る。スタイルの類似性については、所定の分類テーブルを参照することにより、類似したスタイルを関連づける述語によって定義される。本実施例においては、「類似した（"similar"である）」という述語は、以下のように定義されている。

【００４０】

【数６】

let j be the index of the style parameter in the database

in this case, p.aj represents the style of piece p

similar(piece1,piece2,j):=(piece1.aj and piece 2.aj are linked in the

taxonomy of styles)

【００４１】相違制約

上述した類似制約を利用して、逆に、例えば最初から１０番目までといった所定の連続する範囲内の曲は全て属性が異なっていなければならない等の条件を宣言することもできる。この場合、上述の「similar」を用いて、以下のように表現できる。

【００４２】

＊【数４】

$$CA(I, j, card(I), card(I))$$

【００４５】ランニング制約

以下に説明する制約は、連続して発生する２つのアイテム間の関係を拘束する制約であるため、「ランニング」制約と呼ばれる。ランニング制約には類似制約と相違制約という２つの制約クラスがある。

【００４６】類似制約

この制約クラスは、連続するアイテムが所定の範囲内において互いに類似していることを宣言する。この類似性は、所定の属性ｊを拘束する２値述語により定義される。すなわち、類似制約は、インデックスを表す整数をａ及びｂとし、属性のインデックスをｊとして以下のように表現される。

【００４７】

【数５】

$$similar(x, y, j) := (x.a_j \neq y.a_j)$$

【００４３】特別な制約クラスに属する制約を用いた定式化及び解決

上述のような特別な制約クラスを用いることにより、アークコンシステンシの手法を用いることなく、適切且つ完全な解決を行うことができる。すなわち、全ての制約

を充足する全ての解を求めることができる。

【００４４】なお、以下に記述する手順において、変数のドメインに属する値の全てが除外された場合、これは、現在の検索条件においては制約を充足する解が存在しないことを意味する。したがって、ソルバーによる処理では、条件が不成立となり、これがトリガとなって、検索空間の他の部分の検索が開始される。このような手法は、従来から制約充足アルゴリズムが有する特徴である。

＊【００４５】アイテムに関する基礎制約の実現

この制約は、ｉは集合Ｉの要素であるとして、変数X_iを拘束する。この制約は、ａ及びｂを整数として、集合Ｅに属する値の数がａとｂの間の範囲内でなくてはならないことを要求する。この制約は、以下のようなプログラムにより実現される。

【００４６】

【数８】

```
*
Procedure itemCardinality(I, j, a, b, E)
        Integer p, q, r;
        P:={X_i ; i∈I so that ∀x∈domain(X_i), (x.a_j)∈E}
        Q:={X_i ; i∈I so that {x.a_j for x∈domain (Xi)∩E}
        P:=cardP
        q:=cardQ
        if((a<q)or(b>p))then
                the constraint cannot be satisfied
                return a failure handled by the solver
        if(a=q)then
                for every X element of Q do
                        for every x in the domain of X do
                                if E does not contain x,a j  then
                                        remove x from the domain
of X
        if (b=p)then
                for every X element of Q＼Pdo
                        for every x in the domain of X do
                                if E contains x.a j then
                                        remove x from the domain
of X
```

【００４７】属性値に関する基礎制約の実現

この制約は、ｉは集合Ｉの要素であるとして、変数X_iを拘束する。この制約の定義には、［ａ，ｂ］で示される一定の間隔が必要である。この制約は、所定のパラメータｊに関する変数の値の集合、すなわち{value(X1).aj,....value(Xn).aj}がａとｂの間の異なる要素を含むことを要求する。この制約は、以下のようなプログラムにより実現できる。

【００４８】

【数９】

```
Procedure valueCardinality(I,j,a,b)
        A={i∈I so that Xi is instantiated}
        "a variable is instantiated when its domain is a si
ngleton"
        V=[(value(Xi))aj,i in A]
        If(card(V)>b)then
                raise a failure
        If(card(V)=b)then
                for every i∈I\A do
                        for every x in the domain of Xi do
                                if V does not contain x.aj then
                                        remove x from the domain
of Xi
        if(card(I)-card(A)+card(V)<a)then
                raise a failure that is handled by the solver
        if(card(I) - card(A)+card(V)= a)then
                state a new constraint:allDiff(A,j)
                which requires that the variables that are no
t yet
                installed should have different valuos(see be
low)
```

【００４９】類似制約の実現
この制約S([a,b],j,similar(....))は、連続する変数に対して一連の２値的な制約を設定することにより実現できる。類似制約は、以下のようなプログラムにより実現できる。

【００５０】
【数１０】

```
For i:=a to b-1do
        Set constraint[similar(Xi,Xi+1,j)]
```

ここで、similarという制約は以下のように定義される。

```
Procedure filterSimilar(X,Y,j)
        loop 1:for every x in the domain of X do
                boolean keep:=false
                loop 2:for every y in the domain of Y do
                        if(similar(x,y,j))then
                                keep:=true;
                                break loop 2;
                end loop 2
                if(not(keep))then
                        remove x from the domain of X
        end loop 1
        loop 1:for every y in the domain of Y do
                boolean keep:=false
                loop 2:for every x in the domain of X do
                        if(similar(x,y,j))then
                                keep:= true;
                                boolean loop 2;
                end loop 2
                if(not(keep))then
                        remove y from the domain of Y
        end loop 1
```

【００５１】"similar(x,y,j)"という部分はブール値をとるが、それはsimilar(x,y,j)=trueif $x.a_j$ is similar to $y.a_j$．と表すことができ、"breakloop i "という命令はiというラベルが与えられたループを中断することを意味する。

【００５２】基礎制約の解決速度を速めるための冗長的制約の追加

同一の変数の集合に対して複数の基礎制約が宣言されている場合、ソルバーは所定の時間内に解を見つけたり、あるいは適切な時間内に解がないことを明らかにしたりすることができない場合もある。以下、このような場合について説明する。

【００５３】例えば、制約を受ける変数の集合｛X_1，・・・，X_{10}｝が存在し、この集合に対して、以下のような２つの制約が宣言されているとする。

【００５４】１）Cl({1,...,10}1,4,10,{"Rock","country"})。（なお、1は、ファミリーパラメータのインデックスを表す。）

２）Cl({1,...,10}1,4,10,{"Pop""Jazz"})。（なお、この場合ファミリーパラメータのインデックスを表す。）

ソルバーは、制約充足プログラムの標準的なモデルに基づいて、それぞれの制約を個別に検討するため、ソルバーにとって上述のような単純な問題の解決が困難な作業となる。すなわち、標準的なソルバーは、任意に選出した値のうちの最初の６つの値を制約とは無関係にインスタンス化する。これは、第１の制約は、ファミリーパラメータが"Rock"か"Country"のいずれかに相当する楽曲を含むドメインを有する変数が少なくとも４つ以上存在する限りにおいては効果がないということによって説明される。同様なことが、第２の制約についてもいえる。

【００５５】例えば、最初の６つの変数により、対応する楽曲のファミリが"Funk"であることが示されているとする。この場合、第１の制約を充足するためには、残り４つの変数が示すファミリーは、"Rock"か"Country"でなくてはならない。同時に、第２の制約を充足するためには、残りの４つの変数が示すファミリーは、"Pop"か"Jazz"でなくてはならない。この場合、この２つの制約を同時に充足することは、不可能である。

＊【００５６】すなわち、{"Rock","Country"}と{"Pop","Jazz"}とは、共通の構成要素を有しておらず、したがって、ソルバーは、"family"のパラメータが"Rock"か、"Country"か、"Pop"か"Jazz"のいずれかに相当する楽曲を含むドメインを有する変数の残りの数が８つになった時点ですぐに制約の処理を行わなければならない。この問題を解決するために、本実施の形態では、冗長的制約、すなわち、等しい解の集合を得ることができるとともに問題の処理速度を早めることができるような付加的な制約を宣言する。上述した例において付加される冗長的制約は、以下のとおりである。

【００５７】３）CI({1,...,10},1,8,10,{"Rock","Country","Pop","Jazz"})

より一般的に言えば、アイテムに関する基礎制約が宣言されるたびに、以下のような手続きが実行されて、これにより冗長的制約が付加される。

【００５８】

＊【数１１】

```
Let CI(J,j,a',b' F)be the new cardinality constraints on
items.
For every previously-stated cardinality constraint on ite
ms CI(I,I,a,b,E)do
If(I=j)then
        Integer n
        n:=card(E∩F)
        if(a+a'-n>a)and(a+a'-n>a')then
                state constraints:CI(I∪J,i,a+a'-n,b+b',E∪
F)
        if(I<>J)then
                state constraint:
                CI(I∩J,i,a+a'-n-card(I)-card(J)+2.card(I∩
J),b+b',E∪F)
```

【００５９】リサイタル構成装置への適用例

以下では、本発明を適用して実現した、楽曲の連続演奏を目的とするリサイタル構成装置について説明する。この具体例において、サンプルデータベースにはソニー・ミュージックエンタテインメント（Sony Music Entertainment）の曲が収録されている。収録されているアイテムは、整数で表されるインデックス、文字列で表されるタイトル、文字列で表されるアーティスト名、所定のファミリー一覧表に定義されているファミリー、所定のサブファミリー一覧表に定義されているサブファミリー、整数で表される演奏時間、使用されている可能性のあるリズムのタイプ、使用されている可能性のあるテンポ、使用されている可能性のある演奏楽器タイプその１、使用されている可能性のある演奏楽器のタイプその２、採用されている可能性のあるボイスタイプなどの属性等により記述される。

【００６０】この具体例において、ファミリー及びサブファミリーは、音楽スタイルを相互の類似性により関連付けた所定の分類に基づいて定義される。実際のファミリー及びサブファミリーは、「スタイル」という総合的

な属性及びファミリーとサブファミリーの組合せを示す値によって示される。例えば、「ジャズ・スイング」というスタイルは、「ジャズ」というファミリーと「スイング」というサブファミリーを組み合わせたものである。

【００６１】この分類法により、スタイルは相互に関連付けられる。例えば、スタイル「ジャズ・スイング」は、この分類法によりスタイル「ジャズ・クルーナー」と相互に関連付けられ、また、スタイル「ジャズ・クルーナー」は、スタイル「カントリー・クルーナー」と相互に関連付けられる。この分類法により、ｘとｙをスタイルとして、similar(x,y)といった述語を導入することができる。

【００６２】この具体例において、自動的にリサイタルの構成を生成するリサイタル構成装置は、例えば表示装置を備える適切にプログラミングされた汎用コンピュータを用いて実現できる。これによりユーザは、制約の集合を表示装置に表示させ、視覚に基づく操作により所望の制約の集合を特定することができる。また、この汎用コンピュータにより、任意のデータベースに関する所望の制約の解を算出させることができ、さらに、ユーザは、算出された解を表示装置に表示させて視覚的に確認することもできる。

【００６３】ここで、所定のデータベースから楽曲を選出して、以下のような要求を満たす「A gentle path through soul music」というタイトルのリサイタルを構成するシーケンスを生成する場合について検討する。

【００６４】・ＣＤ又はミニディスクに合わせて１２曲からなるシーケンスであること
・スタイルの観点から連続性のあるシーケンスであること（すなわち、いずれの曲も前の曲とスタイルが類似していること。この類似性はスタイルの内在的な分類によって定められる。）
・システムが、より興味深い曲順のリサイタルを構成できるように、最初は「ソウル・ジャズ」で始まり最後は「ソウル・クルーナー」で終わるが、「ソウル・ジャズ」から「ソウル・クルーナー」に直接つながらないようにすること
・テンポの速い曲で始まったりテンポの遅い曲で終わることを避けるために、テンポの遅い曲で始まり、徐々にテンポが早くなってゆくような構成であること
・テンポ順に構成されていること。それぞれの曲のテンポが前の曲のテンポと類似していること（この類似性は、テンポの分類における階層構造的な関係によって決定される。）
・全て異なる曲で構成されていること
・アーティストが曲ごとに異なっていること
音楽的シーケンスに関する以上のような要求は、以下のような制約を有する制約充足問題として忠実に定式化することができる。

【００６５】・最初と最後の曲のスタイルを指定する単純な単項制約
・スタイルの類似性に関するランニング制約
・スタイルに関する互換性のない値の単純なランニング制約
・最初と最後の曲のテンポを指定する単純な単項制約
・テンポの類似性に関するランニング制約
・楽曲すなわちインデックス属性についての包括的な相違制約
・アーティストについての包括的な相違制約
これらの制約は以下のように表される。
・Cl({1},style,1,1{Soul-Jazz})
・Cl({1},style,1,1{Soul-Crooner})
・S([1,12],style, similar(.,...))
・S([1,12]style,different(.,...))
・Cl({1},tempo,0,0{"fast,very fast"})
・Cl({1},tempo,0,0{"slow,very slow"})
・S([1,12],tempo,similar(.,...))
・allDiff([1,12],index)
・allDiff([1,12],authors)
なお、パラメータは、通常インデックス値により示されるが、ここでは、理解を容易にするために、各パラメータをインデックス値ではなくパラメータ自身の名称により示している。

【００６６】本発明の手法を用いて上述の問題を解決することにより得られた楽曲のシーケンスを図１に示す。この図１では、生成されたシーケンスを構成する各楽曲のタイトル、タイトル、アーティスト、演奏時間、テンポ、ボイスタイプ、楽器のタイプ１及び楽器のタイプ２を示している。

【００６７】このように、本発明の手法をリサイタル構成装置に適用することにより、特定の所望の条件を全て満たすリサイタルの構成を自動的に生成することができる。

【００６８】このようなリサイタルの自動生成の技術は、例えば複数のコンパクトディスク（ＣＤ）を装着可能なＣＤプレーヤ等、複数の記録媒体に接続された双方向アービトレーション回路を備える音楽再生装置に組み込まれるモジュールとして実現することもできる。

【００６９】
【発明の効果】以上のように、本発明に係るシーケンス情報生成方法及びシーケンス情報生成装置は、所望のシーケンスにおけるアイテムの属性の値又は値の変域に対する要求により所望のシーケンスの特性を特定し、各アイテムを変数とし、所望のシーケンスについて特定された特性を制約とすることにより、所望のシーケンスの生成を制約充足問題として定式化し、これを解決する。これにより、構成アイテムの類似性や相違性、アイテムに関する任意の分類等に基づいて、所望の基準に適合するシーケンスを効率的に生成することができる。

【図面の簡単な説明】 ＊スを示す図である。

【図１】本発明の手法により生成された楽曲のシーケン＊

【図１】

シーケンス内での順番	タイトル	アーティスト	スタイル（ファミリー＋サブファミリー）	演奏時間	テンポ	ボーカルのタイプ	楽器のタイプ(1, 2)
1	In my life	Souled Out	SoulJazz	256 sec	fast slow	L	シンセサイザ、キーボード
2	Alma de tu flor	Ruben Blades	LatinoJazz	241sec	fast slow	D	パーカッション、アコーディオン
3	Boriqua's anthem	C&C Music Factory	WorldLatino	275 sec	fast	E	パーカッション、ブラス
4	Estoy aqui	Shakira	WorldSoleil	230 sec	fast	I	アコースティックギター、ストリングス
5	Take it easy	Mad Lion	World Reggae	273 sec	fast slow	B	リズムボックス、ピアノ
6	Shine	Aswad	World Reggae	228 sec	fast	M	キーボード、ブラス
7	Brown-eyed girl	Steel Pulse	World Reggae	226 sec	fast	M	キーボード、ブラス
8	Carota Nacional	Skank	World Reggae	243 sec	fast slow	E	ピアノ、ブラス
9	Call me up	Simone Hines	SoulCrooner	289 sec	fast slow	C	ファンクベース、キーボード
10	Are you Jimmy Ray	Jimmy Ray	PopSoul	209 sec	fast	C	ジャズギター、キーボード
11	Low down	Boz Scaggs	PopSoul	318 sec	fast slow	A	フルート、ファンクベース
12	Love so strong	Secret Life	SoulCrooner	270 sec	fast	C	ファンクギター、キーボード